YITP-SB-01-10

Evolution program for parton densities with perturbative heavy flavor boundary conditions

A. Chuvakin, J. Smith
C.N. Yang Institute for Theoretical Physics,
State University of New York at Stony Brook, New York 11794-3840.

February 2001

Abstract

A new code for the scale evolution of modified-minimal-subtraction-scheme parton densities is described. Through next-to-leading order the program uses exact splitting functions. In next-to-next-to-leading order approximate splitting functions are used. For efficiency the program includes analytical results for the evaluation of the weights required for the integrations over the longitudinal momentum fractions of the partons. It also incorporates the operator matrix elements required for the matching conditions across heavy flavor thresholds in higher order perturbation theory. The more efficient handling of the weights implies that the code is faster than similar evolution codes in all modes of operation. The program is written in the C programming language.

PACS numbers: 11.10Jj, 12.38Bx, 13.60Hb, 13.87Ce.

Contents

1 Program Summary

Title of Program: ADENS
Computer: AlphaStation 4/500
Operating system: OSF V3.2
Programming language: C
Number of lines in distributed program: 11658
Keywords: parton density, evolution, numerical solution, splitting function, next-to-next-to-leading order
Nature of physical problem: solution of the parton density evolution equations with LO, NLO and NNLO splitting functions and NLO, NNLO heavy flavor threshold matching conditions
Method of solution: x-space integration with analytic evaluation of weights
Typical running time: see table in Section 7
The program www site:
http://insti.physics.sunysb.edu/∼chuvakin/adens-24.0.tar.gz

http://insti.physics.sunysb.edu/∼smith/adens-24.0.tar.gz

Contents

2 Introduction

Deep-inelastic lepton-hadron scattering experiments probe the internal structure of hadrons. The lepton-hadron inclusive cross sections may be written in terms of structure functions, which depend on the virtuality of the probe Q^2. Three structure functions F_1, F_2 and F_L are necessary to describe neutral current (photon and Z-boson exchange) and charged current (W-boson exchange) reactions. In perturbative quantum chromodynamics (pQCD) the probe interacts with partonic constituents of the hadron. There are probability densities $f(x, \mu^2)$ to find partons carrying a fraction x ($0 < x \leq 1$) of the longitudinal momentum of the hadron at a mass factorization scale μ. Therefore the F_i, $i = 1, 2, L$ also depend on x and μ.

The operator product expansion (OPE) allows the structure functions to be written as convolutions of the parton (quark and gluon) probability densities with partonic hard scattering cross sections (or coefficient functions). The latter can be calculated in pQCD. Even though the former cannot be calculated in pQCD, their μ dependence is determined by a set of integro-differential equations, the (Dokshitzer-Gribov-Lipatov)-Altarelli-Parisi evolution equations [1], which follow from renormalization group analysis. Discussions of the pQCD description of deep inelastic scattering reactions are available in [2] and [3]. The probability densities and splitting functions are defined in the modified-minimal-subtraction ($\overline{\rm MS}$) scheme.

For simplicity we will call the above equations the evolution equations. They describe processes where a massless parton (quark or gluon) carrying a fraction of the longitudinal momentum of the incoming hadron radiates a massless parton and becomes a (different) massless parton with a different momentum fraction. The probability for this process to happen is determined by splitting functions which are computed order-by-order in pQCD. The leading-order (LO) and next-to-leading order (NLO) splitting functions have been known for some time [4], [5], [6], [7], [8], [9] and the results are summarized in a convenient form in [3]. Recently some moments of the next-to-next-to-leading order (NNLO) splitting functions have been calculated in [10], see also [11] and [12] . If the x-dependence of the quark and gluon densities in a hadron are parametrized at one value of μ, (say at μ_0,) then the solutions of the evolution equations with the above LO, NLO or NNLO splitting functions yield the x dependence of the massless parton densities at a different μ. There is a second scale in the pQCD theory, the renormalization scale, which appears in argument of the the running coupling α_s. It is usually set to be the same as the mass factorization scale μ so $\alpha_s = \alpha_s(\mu^2)$.

The flavor dependence of the quark and anti-quark densities is governed by the flavor group, which is SU(2) for the up and down quarks, SU(3) for the up, down and strange quarks etc. Therefore is is convenient to form flavor non-singlet and flavor (pure) singlet combinations of densities. The former have their own evolution equations. The latter mix with the gluons and the combined evolution is described by matrices which obey coupled integro-differential equations.

A number of methods to solve the evolution equations for the parton densities have been proposed, including direct x-space methods, [13], [14], [15], [16], [17], [18], orthogonal polynomial methods [19], [20], and Mellin-transform methods [21], [22]. A compilation of parton density sets is available in [23].

The best method, which should be both accurate and fast, depends on region chosen in x and μ^2. Currently the requirements are that the code be able to evolve densities from a minimum μ^2 near 0.26 GeV^2 up to a maximum μ^2 near 10^6 GeV^2 required for QCD studies for the future Large Hadron Collider at CERN. The range in x is from a minimum value near 10^{-5} up to a maximum near unity. We use the direct x-space method, with the following additional features.

One of our aims is a better treatment of parton density evolution for "light" u, d and s quarks near the heavy flavor thresholds chosen to be at the charm and the bottom quark masses (m_c and m_b respectively). The parton density description must be modified to incorporate new c and b "heavy" quark densities as the evolution scale increases. The implementation of the NLO and NNLO matching conditions across heavy flavor thresholds in the variable flavor number schemes (VFNS) [24], [25], [26], [27] involve large cancellations between various terms in the expressions for the structure functions. Poor numerical accuracy in the solution for the evolution of the parton densities at small scales would spoil these cancellations and ruin the VFNS predictions. We achieve the required accuracy by avoiding one numerical integration in our program so we analytically calculate the weights for the exact LO, the exact NLO and the approximate NNLO splitting functions. The approximate NNLO splitting functions are taken from [28],[29], while the relevant operator matrix elements (OMEs), which provide the matching conditions on the parton densities across heavy flavor thresholds, are taken from [30].

Since we start the scale evolution from a set of densities (input boundary conditions) at a low scale $\mu = \mu_0 \ll m_c$, the running coupling $\alpha_s(\mu^2)$ is large. We therefore use the exact solution of the NLO equation for α_s and match the values on both sides of the heavy flavor thresholds to three decimal places. We mention here that the NNLO matching conditions on α_s across heavy flavor thresholds are available in [31] and [32]. Our program evolves both light and heavy parton densities in LO, NLO and NNLO from a minimum x equal to

10^{-7} to a maximum x equal to unity, a mimimum $\mu^2 = 0.26$ (GeV)^2 in LO and $\mu^2 = 0.40$ (GeV)^2 in NLO and NNLO and and a maximum $\mu^2 = 10^6$ (GeV)^2. Results have been published in [25], [26], [27] and [33]. Here we give a detailed write up of the program.

3 The evolution equations

3.1 Definitions of densities

We evolve combinations of up (u), down (d), strange (s), charm (c) and bottom (b) quark densities which transform appropriately under the flavor group. Hence we define flavor-non-singlet valence quark densities by

$$f_{k-\bar{k}}(n_f, x, \mu^2) \equiv f_k(n_f, x, \mu^2) - f_{\bar{k}}(n_f, x, \mu^2)\,,\; k = u, d\,. \tag{3.1}$$

The flavor-singlet quark densities

$$f_q^{\rm S}(n_f, x, \mu^2) = \sum_{k=1}^{n_f} f_{k+\bar{k}}(n_f, x, \mu^2) \tag{3.2}$$

are defined in terms of the expression

$$f_{k+\bar{k}}(n_f, x, \mu^2) \equiv f_k(n_f, x, \mu^2) + f_{\bar{k}}(n_f, x, \mu^2)\,,\; k = u, d, s, c, b\,, \tag{3.3}$$

when $n_f = 5$. Then the flavor-non-singlet sea quark densities are

$$f_q^{\rm NS}(n_f, x, \mu^2) = f_{k+\bar{k}}(n_f, x, \mu^2) - \frac{1}{n_f} f_q^{\rm S}(n_f, x, \mu^2)\,. \tag{3.4}$$

These equations will be discussed further in the next section.

3.2 The evolution equations

A typical evolution equation is that for a flavor-non-singlet parton density $f^{NS}(x, \mu^2)$

$$\frac{\partial}{\partial \ln \mu^2}\; f^{NS}(y, \mu^2) \;= \frac{\alpha_s(\mu^2)}{2\pi} \int_y^1 \frac{dx}{x}\; P_{NS}(\frac{y}{x}, \mu^2)\; f^{NS}(x, \mu^2) \quad , \tag{3.5}$$

where $P^{NS}(y/x, \mu^2)$ is a non-singlet splitting function, and $\alpha_s(\mu^2)$ is the running coupling.

The splitting functions in the evolution equations can be expanded in a perturbation series in α_s into LO, NLO and NNLO terms as follows

$$P(z,\mu^2) = P^{(0)}(z,\mu^2) + (\frac{\alpha_s(\mu^2)}{2\pi})P^{(1)}(z,\mu^2) + (\frac{\alpha_s(\mu^2)}{2\pi})^2 P^{(2)}(z,\mu^2). \quad (3.6)$$

The non-singlet combinations of the $q_r(\bar{q}_r)$ to $q_s(\bar{q}_s)$ splitting functions, where the subscripts r,s denote the flavors of the (anti)quarks q and $\bar{q}$ respectively and satisfy $r,s = 1,\cdots,n_f$, can be further decomposed into flavor diagonal parts proportional to δ_{rs} and flavor independent parts. In LO there is only one non-singlet splitting function P_{qq} but in NLO it is convenient to form two combinations from P_{qq} and $P_{q\bar{q}}$ as follows

$$\begin{aligned} P_+ &= P_{qq} + P_{q\bar{q}}\,, \\ P_- &= P_{qq} - P_{q\bar{q}}\,. \end{aligned} \quad (3.7)$$

These splitting functions are used to evolve two independent types of non-singlet densities, which will be called plus and minus respectively. They are given by

$$\begin{aligned} f_i^+ &= f_q^{\rm NS}(n_f,x,\mu^2)\,, \\ f_j^- &= f_{k-\bar{k}}(n_f,x,\mu^2)\,. \end{aligned} \quad (3.8)$$

Since the general formulae in Eqs. (3.1)-(3.4) are rather involved the easiest way to explain the indices is by explicitly giving the combinations we use. For $j = 1,2$ we have

$$f_1^- = u - \bar{u}\,,\; f_2^- = d - \bar{d}\,, \quad (3.9)$$

which are used for all flavor density sets. Then for three-flavor densities $i = 1,2,3$ and we define

$$\begin{aligned} f_1^+ &= u + \bar{u} - \Sigma(3)/3\,, \qquad f_2^+ = d + \bar{d} - \Sigma(3)/3\,, \\ f_3^+ &= s + \bar{s} - \Sigma(3)/3\,, \end{aligned} \quad (3.10)$$

where $\Sigma(3) = f_q^S(3) = u + \bar{u} + d + \bar{d} + s + \bar{s}$. These densities should be used for scales $\mu < m_c$. For four-flavor densities $i = 1,2,3,4$ and we define

$$\begin{aligned} f_1^+ &= u + \bar{u} - \Sigma(4)/4\,, \qquad f_2^+ = d + \bar{d} - \Sigma(4)/4\,, \\ f_3^+ &= s + \bar{s} - \Sigma(4)/4\,, \qquad f_4^+ = c + \bar{c} - \Sigma(4)/4\,, \end{aligned} \quad (3.11)$$

where $\Sigma(4) = f_q^S(4) = c + \bar{c} + \Sigma(3)$. These should be used for scales in the region $m_c \le \mu < m_b$. For five-flavor densites $i = 1,2,3,4,5$ and we define

$$
\begin{aligned}
f_1^+ &= u + \bar{u} - \Sigma(5)/5\,, \qquad f_2^+ = d + \bar{d} - \Sigma(5)/5\,,\\
f_3^+ &= s + \bar{s} - \Sigma(5)/5\,, \qquad f_4^+ = c + \bar{c} - \Sigma(5)/5\,,\\
f_5^+ &= b + \bar{b} - \Sigma(5)/5\,,
\end{aligned}
\tag{3.12}
$$

where $\Sigma(5) = f_q^S(5) = b + \bar{b} + \Sigma(4)$. These should be used for scales $\mu \geq m_b$.

If we define $t = \ln(\mu^2/(1\,\mathrm{GeV}^2)$ then we need to solve the four evolution equations

$$
\frac{\partial f_i^+(y,t)}{\partial t} = \frac{\alpha_s(t)}{2\pi} \int_y^1 \frac{dx}{x} P_+(\frac{y}{x},t) f_i^+(x,t)\,,
\tag{3.13}
$$

$$
\frac{\partial f_j^-(y,t)}{\partial t} = \frac{\alpha_s(t)}{2\pi} \int_y^1 \frac{dx}{x} P_-(\frac{y}{x},t) f_j^-(x,t)\,,
\tag{3.14}
$$

$$
\frac{\partial f_g(y,t)}{\partial t} = \frac{\alpha_s(t)}{2\pi} \int_y^1 \frac{dx}{x} \left[P_{gq}(\frac{y}{x},t) f_q^S(x,t) + P_{gg}(\frac{y}{x},t) f_g^S(x,t) \right],
\tag{3.15}
$$

$$
\frac{\partial f_q^S(y,t)}{\partial t} = \frac{\alpha_s(t)}{2\pi} \int_y^1 \frac{dx}{x} \left[P_{qq}(\frac{y}{x},t) f_q^S(x,t) + P_{qg}(\frac{y}{x},t) f_g^S(x,t) \right],
\tag{3.16}
$$

where for $\mu < m_c$ we set $i = 1,2,3$, $j = 1,2$, $f_q^S = \Sigma(3)$ and the gluon is a three-flavor gluon. When $m_c \leq \mu < m_b$, we use $i = 1,2,3,4$, $j = 1,2$, $f_q^S = \Sigma(4)$ and the gluon is a four-flavor gluon. Finally when $\mu \geq m_b$, we set $i = 1,2,3,4,5$, $j = 1,2$, $f_q^S = \Sigma(5)$ and the gluon is a five-flavor gluon. Note that since NNLO splitting functions are approximate we provide the high and low estimate for each splitting functions labeled A and B. For all calculations we use their average so that the error is minimized.

The densities satisfy the momentum conservation sum rule which we write in terms of the $u, d, ..b$ (anti)-quark and gluon densities as

$$
\begin{aligned}
&\int_0^1 dx\, x \Big[u(x,\mu^2) + \bar{u}(x,\mu^2) + d(x,\mu^2) + \bar{d}(x,\mu^2)\\
&+ s(x,\mu^2) + \bar{s}(x,\mu^2) + [c(x,\mu^2) + \bar{c}(x,\mu^2)]\theta(\mu^2 - m_c^2)
\end{aligned}
$$

$$+[b(x,\mu^2)+\bar{b}(x,\mu^2)]\theta(\mu^2-m_b^2)+g(x,\mu^2)\Big] \;=1\,. \tag{3.17}$$

Also the quark constituents carry all the charge, isospin, strangeness, charm and bottom quantum numbers of the nucleon so they also satisfy the other standard sum rules for the conservation of these quantities, see [2], [3].

4 Direct x-space method of solution and initial conditions

4.1 The method

Our choice of the direct x-space method is motivated by the necessity to step densities across heavy flavor thresholds using LO, NLO and NNLO boundary conditions. The procedure of doing this with Mellin moments would involve taking moments of the densities and then inverting moments several times. The direct x-space method is much more intuitive and straightforward. The main features of this method are linear interpolation over a grid in x and second-order interpolation over a grid in t. Let us describe it in more detail to point out where we differ from the method in [14].

First we consider the x-variable in the evolution and write the right-hand-side of the evolution equation Eq.(3.5) for the non-singlet density as

$$I(x_0) = \int \frac{dx}{x} \frac{x_0}{x} P\left(\frac{x_0}{x}\right) q\left(x\right) \,, \tag{4.1}$$

where $x_0 \leq x \leq 1$,

$$q(x) = xf(x) \,, \tag{4.2}$$

and

$$x_0 < x_1 < ... < x_n < x_{n+1} \equiv 1 \,, \tag{4.3}$$

with $q(x_{n+1}) = q(1) \equiv 0$. Between grid points x_i and x_{i+1}, x is chosen so that

$$q(x) = (1 - \bar{x})q(x_i) + \bar{x}q(x_{i+1}) \,, \tag{4.4}$$

with $\bar{x} = (x - x_i)/(x_{i+1} - x_i)$. Using this relation we convert the integral into a sum

$$I(x_0) = \sum_{i=0}^{n+1} w(x_i, x_0) q(x_i) \,, \tag{4.5}$$

where the weights are (in all orders LO, NLO and NNLO)

$$\begin{aligned} w(x_0, x_0) &= S_1(s_1, s_0) \\ w(x_i, x_0) &= S_1(s_{i+1}, s_i) - S_2(s_i, s_{i-1}) \,, \end{aligned} \tag{4.6}$$

where $s_i = x_0/x_i$ and

$$S_1(u,v) = \frac{v}{v-u}\int_u^v (z-u)P(z)\frac{dz}{z}\,,$$
$$S_2(u,v) = \frac{u}{v-u}\int_u^v (z-v)P(z)\frac{dz}{z}\,. \tag{4.7}$$

In the above formula $P(z)$ denotes the splitting function of the corresponding order in α_s and type (non-singlet, singlet, etc.) We use the LO and NLO splitting functions in [19] and the approximations to the NNLO splitting functions from [28] and [29]. For completeness the latter are given in Appendix A. We have calculated the integrals in Eq.(4.7) analytically and the results are in the computer program. This yields the formula in Eq.(4.5) describing the grid for the x variable. Note that the weights $w^{(0)}$, $w^{(1)}$ and $w^{(2)}$ are those for the exact LO, the exact NLO and the approximate NNLO splitting functions respectively. Thus, for the singlet case, we have

$$\begin{aligned}\frac{d(x_0\Sigma(x_0))}{dt} = \frac{\alpha_s}{2\pi}\sum_{i=0}^{n+1}\Big[&[w_{qq}^{(0)}(x_i,x_0) + \frac{\alpha_s}{2\pi}w_{qq}^{(1)}(x_i,x_0) + (\frac{\alpha_s}{2\pi})^2 w_{qq}^{(2)}(x_i,x_0)]\\ &\times\ x_i\Sigma(x_i)\\ &+[w_{qg}^{(0)}(x_i,x_0) + \frac{\alpha_s}{2\pi}w_{qg}^{(1)}(x_i,x_0) + (\frac{\alpha_s}{2\pi})^2 w_{qg}^{(2)}(x_i,x_0)]\\ &\times\ x_i g(x_i)\Big]\,,\end{aligned} \tag{4.8}$$

where Σ is either $\Sigma(3)$, $\Sigma(4)$ or $\Sigma(5)$ depending on the scale.

Now consider the variation in the variable t. For each x_i we pick a grid in t labelled by distinct points t_j. Then, for example, the non-singlet equation becomes

$$\begin{aligned}q^{'}(x_i,t_j) = \frac{\alpha_s(t_j)}{2\pi}\sum_{k=1}^{n}&[w_{\pm}^{(0)}(x_k,x_i) + \frac{\alpha_s(t_j)}{2\pi}w_{\pm}^{(1)}(x_k,x_i)\\ &+ (\frac{\alpha_s(t_j)}{2\pi})^2 w_{\pm}^{(2)}(x_k,x_i)]q(x_k,t_j)\,,\end{aligned} \tag{4.9}$$

where $q^{'}(x_i,t_j)$ denotes the derivative with respect to t evaluated at $t=t_j$. In compact notation this equation can be rewritten as

$$q_j^{'} = wq_j + S\,, \tag{4.10}$$

with S being the sum of the terms on the right hand side of Eq.(4.9) excluding

the j-th term.

For t between the grid points t_{j-1} and t_j we interpolate the parton density using quadratic interpolation as follows:

$$q(x_i, t) = at^2 + bt + c\,. \tag{4.11}$$

Thus we relate the value of q at the point t_j to that of q at the point t_{j-1} by

$$q(x_i, t_j) = q(x_i, t_{j-1}) + \frac{1}{2}[q^{'}(x_i, t_j) + q^{'}(x_i, t_{j-1})]\Delta t_j\,, \tag{4.12}$$

where $\Delta t_j = t_j - t_{j-1}$. This equation can also be written more compactly as

$$q_j = q_{j-1} + \frac{1}{2}(q^{'}_{j-1} + q^{'}_j)\Delta t_j\,. \tag{4.13}$$

The resulting system of two linear equations in Eq.(4.10) and Eq. (4.13) for q_j and $q^{'}_j$ has the solution

$$q_j = \frac{2q_{j-1} + (q^{'}_{j-1} + S)\Delta t_j}{2 - w\Delta t_j}\,. \tag{4.14}$$

Then we find $q^{'}_j$ from Eq.(4.10). Applying the same procedure to the gluon and singlet combinations involves four equations because we have to compute both the densities and their derivatives.

The evolution proceeds from the initial $\mu_0^2 = \mu_{\rm LO}^2$ (or $\mu_0^2 = \mu_{\rm NLO}^2$) to the first heavy flavor threshold at the scale $\mu^2 = m_c^2$. Next the charm density is introduced in NNLO (α_s^2-order terms) and all the four-flavor densities are evolved from the new boundary conditions in Section 4.2. This evolution continues up to the transition point $\mu^2 = m_b^2$, where the same procedure is applied to generate the bottom quark density. From that matching point all five-flavor densities are evolved up to all higher μ^2 scales starting from the boundary conditions in Appendix B.

Since the weights for the calculation are computed analytically from the LO, NLO [19] and NNLO ([28],[29]) $\overline{\rm MS}$ splitting functions we remove possible instabilities in the numerical integrations. Hence the program is very efficient and fast. The results from the evolution code have been thoroughly checked against the tables in the HERA report [16] and they agree to all five decimal places.

The GRV98 [22] three-flavor LO and NLO parton density sets contain input formulae at low scales $\mu < m_c$ which are ideal as initial values for our parametrizations. Therefore we start our LO evolution using the following input at $\mu_0^2 = \mu_{\rm LO}^2 = 0.26$ GeV^2

$$\begin{aligned}
xf_{u-\bar{u}}(3,x,\mu_0^2) &= xu_v(x,\mu_{\rm LO}^2) \\
&= 1.239\ x^{0.48}\,(1-x)^{2.72}\,(1-1.8\sqrt{x}+9.5x) \\
xf_{d-\bar{d}}(3,x,\mu_0^2) &= xd_v(x,\mu_{\rm LO}^2) \\
&= 0.614\ (1-x)^{0.9}\ xu_v(x,\mu_{\rm LO}^2) \\
x(f_{\bar{d}}(3,x,\mu_0^2) - f_{\bar{u}}(3,x,\mu_0^2)) &= x\Delta(x,\mu_{\rm LO}^2) \\
&= 0.23\ x^{0.48}\,(1-x)^{11.3}\,(1-12.0\sqrt{x}+50.9x) \\
x(f_{\bar{d}}(3,x,\mu_0^2) + f_{\bar{u}}(3,x,\mu_0^2)) &= x(\bar{u}+\bar{d})(x,\mu_{\rm LO}^2) \\
&= 1.52\ x^{0.15}\,(1-x)^{9.1}\,(1-3.6\sqrt{x}+7.8x) \\
xf_g(3,x,\mu_0^2) &= xg(x,\mu_{\rm LO}^2) \\
&= 17.47\ x^{1.6}\,(1-x)^{3.8} \\
xf_s(3,x,\mu_0^2) = xf_{\bar{s}}(3,x,\mu_0^2) &= xs(x,\mu_{\rm LO}^2) \\
&= x\bar{s}(x,\mu_{\rm LO}^2) = 0\,. \qquad (4.15)
\end{aligned}$$

Here $\Delta \equiv \bar{d} - \bar{u}$ is used to construct the non-singlet combination.

We start the corresponding NLO evolution using the following GRV98 input at $\mu_0^2 = \mu_{\rm NLO}^2 = 0.40$ GeV^2

$$\begin{aligned}
xf_{u-\bar{u}}(3,x,\mu_0^2) &= xu_v(x,\mu_{\rm NLO}^2) \\
&= 0.632\ x^{0.43}\,(1-x)^{3.09}\,(1+18.2x) \\
xf_{d-\bar{d}}(3,x,\mu_0^2) &= xd_v(x,\mu_{\rm NLO}^2) \\
&= 0.624\ (1-x)^{1.0}\ xu_v(x,\mu_{\rm NLO}^2) \\
x(f_{\bar{d}}(3,x,\mu_0^2) - f_{\bar{u}}(3,x,\mu_0^2)) &= x\Delta(x,\mu_{\rm NLO}^2) \\
&= 0.20\ x^{0.43}\,(1-x)^{12.4}\,(1-13.3\sqrt{x}+60.0x) \\
x(f_{\bar{d}}(3,x,\mu_0^2) + f_{\bar{u}}(3,x,\mu_0^2)) &= x(\bar{u}+\bar{d})(x,\mu_{\rm NLO}^2) \\
&= 1.24\ x^{0.20}\,(1-x)^{8.5}\,(1-2.3\sqrt{x}+5.7x) \\
xf_g(3,x,\mu_0^2) &= xg(x,\mu_{\rm NLO}^2) \\
&= 20.80\ x^{1.6}\,(1-x)^{4.1} \\
xf_s(3,x,\mu_0^2) = xf_{\bar{s}}(3,x,\mu_0^2) &= xs(x,\mu_{\rm NLO}^2) \\
&= x\bar{s}(x,\mu_{\rm NLO}^2) = 0. \qquad (4.16)
\end{aligned}$$

We start the corresponding NNLO evolution using the same NLO input and starting scale as above.

4.3 The calculation of the running coupling

The heavy quark masses $m_c = 1.4$ GeV^2, $m_b = 4.5$ GeV^2 are used throughout the calculation. We also use the exact solution (as opposed to a perturbative solution in inverse powers of $\ln(\mu^2/\Lambda^2)$) of the differential equation

$$\frac{d\,\alpha_s(\mu^2)}{d\,\ln(\mu^2)} = -\frac{\beta_0}{4\pi}\,\alpha_s^2(\mu^2) \;-\; \frac{\beta_1}{16\pi^2}\,\alpha_s^3(\mu^2)\,, \tag{4.17}$$

for the running coupling $\alpha_s(\mu^2)$. Here $\beta_0 = 11 - 2n_f/3$ and $\beta_1 = 102 - 38n_f/3$. Another way of writing this equation is

$$\ln\frac{\mu^2}{(\tilde{\Lambda}^{(n_f)}_{\rm EXACT})^2} = \frac{4\pi}{\beta_0\alpha_s(\mu^2)} \;-\; \frac{\beta_1}{\beta_0^2}\,\ln\left[\frac{4\pi}{\beta_0\alpha_s(\mu^2)} \;+\; \frac{\beta_1}{\beta_0^2}\right]. \tag{4.18}$$

The values for $\tilde{\Lambda}^{(n_f)}_{\rm EXACT}$ are carefully chosen to obtain accurate matching of α_s at the scales m_c^2 and m_b^2. We used the values $\tilde{\Lambda}^{(3,4,5,6)}_{\rm EXACT} = 299.4,\ 246,\ 167.7,\ 67.8$ MeV$/c^2$ respectively in the exact formula (which yields $\alpha_s^{\rm EXACT}(m_Z^2) = 0.114$, $\alpha_s^{\rm EXACT}(m_b^2) = 0.205$, $\alpha_s^{\rm EXACT}(m_c^2) = 0.319$, $\alpha_s^{\rm EXACT}(\mu^2_{\rm NLO}) = 0.578$) and $\Lambda^{(3,4,5,6)}_{\rm LO} = 204,\ 175,\ 132, 66.5$ MeV$/c^2$ respectively (which yields $\alpha_s^{\rm LO}(m_Z^2) = 0.125$, $\alpha_s^{\rm LO}(m_b^2) = 0.232$, $\alpha_s^{\rm LO}(m_c^2) = 0.362$, $\alpha_s^{\rm LO}(\mu^2_{\rm LO}) = 0.763$) for the LO formula (where $\beta_1 = 0$). There is a NNLO discontinuity of aproximately two parts in one thousand in the running coupling across heavy flavor thresholds [31], [32]. We have ignored this effect to focus on the numerically more significant matching of the flavor densities.

4.4 The evolution process

Three flavor evolution proceeds from the initial μ_0^2 to the scale $\mu^2 = m_c^2 = 1.96$ (GeV^2)^2. At this point the charm density is then defined by

$$\begin{aligned} f_{c+\bar{c}}(n_f+1, m_c^2) = a_s^2(n_f, m_c^2)\Big[&\tilde{A}^{\rm PS}_{Qq}(1) \otimes f_q^{\rm S}(n_f, m_c^2) \\ &+\tilde{A}^{\rm S}_{Qg}(1) \otimes f_g^{\rm S}(n_f, m_c^2)\Big]\,, \end{aligned} \tag{4.19}$$

with $n_f = 3$ and $a_s = \alpha_s/4\pi$. We have suppressed the x dependence to make the notation more compact. The $\otimes$ symbol denotes the convolution integral

$f \otimes g = \int f(x/y)g(y)dy/y$, where $x \leq y \leq 1$. The OME's $\tilde{A}^{\rm PS}_{Qq}(\mu^2/m_c^2)$, $\tilde{A}^{\rm S}_{Qg}(\mu^2/m_c^2)$ are given in [30] and are also listed in Appendix B. The reason for choosing the matching scale μ at the mass of the charm quark m_c is that all the $\ln(\mu^2/m_c^2)$ terms in the OME's vanish at this point leaving only the nonlogarithmic pieces in the order α_s^2 OME's to contribute to the right-hand-side of Eq.(4.19). Hence the LO and NLO charm densities vanish at the scale $\mu = m_c$. The NNLO charm density starts off with a finite x-dependent shape in order α_s^2. Note that we then order the terms on the right-hand-side of Eq. (4.19) so that it contains a product of NLO OME's and LO parton densities. The result is then of order α_s^2 and should be multiplied by order α_s^0 coefficient functions when forming the deep inelastic structure functions.

The four-flavor gluon density is also generated at the matching point in the same way. At $\mu = m_c$ we define

$$\begin{aligned} f_g^{\rm S}(n_f+1,m_c^2) = & f_g^{\rm S}(n_f,m_c^2) \\ & + a_s^2(n_f,m_c^2)\Big[A^{\rm S}_{gq,Q}(1) \otimes f_q^{\rm S}(n_f,m_c^2)\,, \\ & + A^{\rm S}_{gg,Q}(1) \otimes f_g^{\rm S}(n_f,m_c^2)\Big]\,. \end{aligned} \tag{4.20}$$

The OME's $A^{\rm S}_{gq,Q}(\mu^2/m_c^2)$, $A^{\rm S}_{gg,Q}(\mu^2/m_c^2)$ are given in [30] and are also listed in the Appendix B. The four-flavor light quark (u,d,s) densities are generated using

$$\begin{aligned} f_{k+\bar{k}}(n_f+1,m_c^2) = & f_{k+\bar{k}}(n_f,m_c^2) \\ & + a_s^2(n_f,m_c^2) A^{\rm NS}_{qq,Q}(1) \otimes f_{k+\bar{k}}(n_f,m_c^2)\,. \end{aligned} \tag{4.21}$$

The OME $A^{\rm NS}_{qq,Q}(\mu^2/m_c^2)$ is given in [30] (as well as in Appendix B) and the *total* four-flavor singlet quark density folows from the sum of Eqs. (4.19) and (4.21). In Eqs. (4.20) and (4.21) we set $n_f = 3$. The remarks after Eq. (4.19) are relevant here too.

Next the resulting four-flavor densities are evolved using the four-flavor weights in either LO, NLO and NNLO up to the scale $\mu^2 = m_b^2 = 20.25\ ({\rm GeV}^2)^2$. The bottom quark density is then generated at this point using

$$\begin{aligned} f_{b+\bar{b}}(n_f+1,m_b^2) = & a_s^2(n_f,m_b^2)\Big[\tilde{A}^{\rm PS}_{Qq}(1) \otimes f_q^{\rm S}(n_f,m_b^2) \\ & + \tilde{A}^{\rm (S)}_{Qg}(1) \otimes f_g^{\rm S}(n_f,m_b^2)\Big]\,, \end{aligned} \tag{4.22}$$

and the gluon and light quark densities (which now include charm) are generated using Eqs.(4.19)-(4.21) with $n_f = 4$ and replacing m_c^2 by m_b^2. Therefore only the nonlogarithmic terms in the order a_s^2 OME's contribute to the matching conditions on the bottom quark density. Then all the densities are evolved

up to higher μ^2 as a five-flavor set with either LO, NLO and NNLO splitting functions. This is valid until $\mu = m_t \approx 175$ GeV^2 above which one should switch to a six-flavor set. We do not implement this step because the top quark density would be extremely small.

The procedure outlined above generates a full set of parton densities (gluon, singlet, non-singlet light and heavy quark densities,) for any x and μ^2 from the three-flavor LO, NLO and NNLO inputs in Eqs.(4.15) and (4.16). Note that we have used μ^2 for the factorization and renormalization scales in the above discussion. In the computer program we use the notation that Q^2 denotes these scales, since this is done in all the previous computer codes for the parton densities.

5 Input parameter description and usage

To prepare the program for use unpack the distribution package **adens-24.tar.gz** by typing tar -xzf adens-24.tar.gz. The resulting directory will contain the following files

```
head.h
main.h
main.c
l-a-w.c
nl-a-w.c
alpha.c
init.c
polylo.c
intpol.c
evolver.c
thresh.c
a-coefs.c
loader.c
quadrat.c
daind.c
integrands.c
grids.c
weights.c
nnl-a-w.c
wgplg.c
evolution_parameters.input
makefile
my_howto.tex
sample.out
```

To build the executable on a machine with a gcc compiler type *make* . The executable named **adens.x** will be produced. To run the code just run the file **adens.x**. Some debugging information may appear on the standard output.

Here is the parameter file (**evolution_parameters.input**) explanation with default values shown:

0.204e0	LambdaLO-3	LO Λ for N_f =3
0.175e0	LambdaLO-4	LO Λ for N_f =4
0.132e0	LambdaLO-5	LO Λ for N_f =5
0.306e0	LambdaNLO3	NLO Λ for N_f =3
0.257e0	LambdaNLO4	NLO Λ for N_f =4
0.1734e0	LambdaNLO5	NLO Λ for N_f =5
0.2994e0	LambdaENLO3	Exact Λ for N_f =3
0.246e0	LambdaENLO4	Exact Λ for N_f =3
0.1677e0	LambdaENLO5	Exact Λ for N_f =3
0.40e0	Qinitial2	Initial Q^2 to start evolution
1.96e0	QcharmMass	Mass of first heavy quark c
20.25e0	QbottomMass	Mass of second heavy quark b
1.96e0	QcharmThreshold	Charm threshold
1.96e0	AlphaCharmThreshold	C threshold used for α_s
20.25e0	QbottomThreshold	Bottom threshold
20.25e0	AlphaQbottomThreshold	B threshold used for α_s
1000.0e0	Qfinal2	Final Q^2
130	tGridSize	Q^2 grid size
200	xGridSize	x grid size
130	xGridSplit	x split between log and linear
1.0e-5	xInitial	x initial
0.2e0	xSplit	x at the split btw log and linear
1.00e0	xFinal	x final (always 1)
0	DebugLevel	Error message detail (0-5)
1	GraphVsX	Plotting data files are versus x (1) or Q^2 (0)
1	Order	LO/NLO/NNLO for 0,1,2
0	DoFortran	Produce (1) or no (0) data files for CSN/BMSN Fortran programs (1-yes, 0-no)
1	AlphaDoSeparateThreshold	Use separate thresholds for α_s (1-yes, 0-no)

1	AlphaUseExact	Use exact GRV98-style α_s (1-yes, 0-no)
0	ThreeFlavorMode	Calculate GRV98-style densities with no heavy flavors (1-yes, 0-no)
0	GraphAll	Plot all data points (1-yes, 0-no)
0	NNLOmultiOrderCHARM	Use our proper order NNLO heavy flavors (1-yes, 0-no)
1	DoBottomThreshold	Generate bottom (1-yes, 0-no)
0	LoadWeightsMadeBefore	Use ready weights if available (1-yes, 0-no)
1	DoNotDumpWeights	Dump weight for future use as the option above (1-yes, 0-no)
0	NLO4NNLO	Use NLO weights for NNLO calculation (1-yes, 0-no)

The first set of Lambdas are used for LO calculations. The second set are used for NLO and NNLO calculations if the exact α_s is not requested (AlphaUseExact=0). The next set (LambdaENLO3, LambdaENLO4, LambdaENLO5) are used for the exact solution of the differential equation for α_s as proposed in the GRV98 paper [22]. The code that calculates the exact α_s might use its own set of flavor thresholds (which means that the number of flavors used for α_s can be reset independently from the regular heavy flavor threshold as done in [22]).

Next we give the Q^2 limits and the heavy masses: the initial and final Q^2, the charm and bottom masses (used in threshold calculations), the heavy flavor thresholds and the separate α_s thresholds. Next follow the grid sizes in x and Q^2 together with x initial and final (always 1) and the switch point between logarithmic and linear grids in the x dimension. The x grid always starts as logarithmic and then becomes linear at higher x, usually at a value of the order of 0.1 (xGridSplit parameter).

The last group of parameters contains various control values that set the modes of the computation:
DebugLevel , controls the amount of generated error, warning and information messages,
GraphVsX , controls the printing of the output data for plotting (first column is either x or Q^2, then subsequent columns will contain density values for various Q^2 or x),

Order, sets calculation order (use 0,1,2 for LO,NLO,NNLO),
DoFortran, sets whether to dump interpolated densities on a special grid for future use in Fortran code for the calculation of structure functions ; CSN and BMSN refer to VFNS schemes which are explained in [25],
AlphaDoSeparateThreshold, sets whether we use a separate threshold for α_s (used, for instance for GRV98 set where n_f for densities is always 3 and n_f for α_s goes from 3 to 5,
AlphaUseExact, sets whether to use exact (differential equation solution) α_s for NLO and NNLO calculation,
ThreeFlavorMode, sets whether to run GRV98 mode (no heavy flavors, $n_f = 3$ for all Q^2),
GraphAll, controls the amount of graphing and printing output (either all data points or the special grid defined in the file **main.h**, that contains some favorite values (for more see Section 7)),
NNLOmultiOrderCHARM, activates NNLO threshold calculation using proper order combinations (this mode requires one to first run the LO and NLO calculations),
DoBottomThreshold, enables the bottom density,
LoadWeightsMadeBefore, turns on and off the loading of weights computed in the prior runs,
DoNotDumpWeights, sets whether to save computed weights to disk for future use,
NLO4NNLO, sets whether NLO weights are used for the NNLO calculation (thus having only the boundary condition in NNLO).

Some common parameter settings and typical grid sizes for popular evolutions are shown in Section 7.

6 Description of the program

6.1 Program module summary

main.c	The main program, input and output
l-a-w.c	Calculation of LO weights
nl-a-w.c	Calculation of NLO weights
alpha.c	Calculation of α_s
init.c	Definition of initial functions
polylo.c	Calculation of polylogarithms
intpol.c	Interpolation routine
evolver.c	Evolution process subroutine
thresh.c	Threshold handling subroutine
a-coefs.c	OMEs for thresholds
loader.c	Datafile reading subroutine
quadrat.c	Gaussian integration subroutine
daind.c	Another integration subroutine
integrands.c	Heavy flavor integrand calculation routine
grids.c	Grid generation routine and memory management routines
weights.c	Weight table handling routine
nnl-a-w.c	Calculation of NNLO weights
wgplg.c	Calculation of high order polylogarithms

6.2 main.c

subroutines:
none.

The main program module contains input handling from the parameter file, parameter verification, calls to grid generating routines (**MakeXGrid, MakeT-**

Grid), resets for all density arrays (array q) and their derivatives (array qp). It also includes calls to the generation of weights (**analowgts, ananlowgts**), the calls to evolution and threshold routines (**evolver, threshold**) that do the actual work. Also it contains some pre-output density processing and the results provided in various formats for both viewing and plotting.

6.3 l-a-w.c

subroutines:
int analowgts(int nf,int loadWgts),
int computeLOwgts(int nf),
double sqq(double x,double y),
double sgg(double x,double y).

Analytically computes or reads from the file the LO weights for the evolution equations.

6.4 nl-a-w.c

subroutines:
int ananlowgts(int nf,int loadWgts),
int computeNLOwgts(int nf),
double s1ff(double x,double y, int nf),
double s2ff(double x,double y, int nf),
double s1fg(double x,double y, int nf),
double s2fg(double x,double y, int nf),
double s1gf(double x,double y, int nf),
double s2gf(double x,double y, int nf),
double s1gg(double x,double y, int nf),
double s2gg(double x,double y, int nf),
double s1ff_plus(double x, int nf),
double s1gg_plus(double x, int nf),
double s1p(double x,double y, int nf),
double s2p(double x,double y, int nf),
double s1m(double x,double y, int nf),
double s2m(double x,double y, int nf),
double s1p_plus(double x, int nf),
double s1m_plus(double x, int nf),
double s1gf_lim(double sp,double nf),
double s1fg_lim(double sp,double nf),
double s2ff_lim(double sp,double nf),

double s2fg_lim(double sp,double nf),
double s2gf_lim(double sp,double nf),
double s2gg_lim(double sp,double nf),
double s2p_lim(double sp,double nf),
double s2m_lim(double sp,double nf).

Analytically computes or reads from the file the NLO weights for the evolution equation. These routines are grouped into 3 kinds: the s1,2xx routines calculate the regular weights, the s1,2xx_lim routines calculate the regular weights called at 1 and s1,2xx_plus do the weights that contain the plus-distributions.

6.5 alpha.c

subroutines:
double alpha(double tt, int nf), double alphae (double tt,int nf).

Calculates LO, NLO and exact running coupling α_s using corresponding parameters from the input file.

6.6 init.c

subroutines:
double initq_uv(double xx),
double initq_dv(double xx),
double init_gl(double xx),
double initq_ss(double xx),
double initq_del(double xx),
double initq_udbar(double xx).

Sets initial values for all parton densities using the GRV98 input for LO and NLO densities from [22].

6.7 polylo.c

subroutines:
double Li2(double x),

double Li3(double x),
double S12(double x).

Calculates these three polylogarithms using a fast routine with Bernouilli numbers. .

6.8 intpol.c

subroutines:
double int_q(int j,double xx,int it),
double interpolate(double xx,double *xt, double *yt,int points).

Interpolation routines used to calculate densities between grid points and for integration at the threshold.

6.9 evolver.c

subroutines:
evolver(int it1,int it2,int ic,int ib).

The main routine that performs the evolution between thresholds for all densities. It updates the main density array q and the density derivatives array qp.

6.10 thresh.c

subroutines:
int threshold(int what,int itt),
int fdens4(double xx,int ittc,double *u,double *d,double *s),
double light_charm(double xx,int ittc),
double fcharm(double xx,int ittc),
double fbottom(double xx,int ittc),
double fsigma(double xx,int ittc),
double fgluon(double xx,int ittc),
double fcharm(double xx,int ittc),
double fbottom(double xx,int ittc).

Threshold handling routines to implement LO, NLO and NNLO matching conditions for light and heavy densities at the charm and bottom thresholds. The density routines are calls to convolution integrals that generate new densities for $n_f + 1$ flavors.

6.11 a-coefs.c

subroutines:
double a1qg(double z,double fs2,double hm2),
double a2qq(double z,double fs2,double hm2),
double a2qg(double z,double fs2,double hm2),
double a2qqns(double z,double fs2,double hm2),
double softq(double z,double fs2,double hm2),
double corq(double z,double fs2,double hm2),
double a2gg(double z,double fs2,double hm2),
double softg(double z,double fs2,double hm2),
double corg1(double fs2,double hm2),
double corg2(double z,double fs2,double hm2),
double a2gq(double z,double fs2,double hm2).

The OME routines used for NNLO threshold matching. These contain the formulae in Appendix B.

6.12 loader.c

subroutines:
int loadOrd(int what).

Functions to handle threshold datafile loading, saving and verification. This file allows one the ability to use previously computed density values at the threshold in a new computation.

6.13 quadrat.c

subroutines:
double qadrat(double *x, double a, double b, double (*fx)(double), double e[]),
double lint(double *x, double (*fx)(double), double e[], double x0, double xn, double f0, double f2, double f3, double f5, double f6, double f7, double f9, double f14, double hmin, double hmax, double re, double ae).

Backup integration routine used as a check for the actual one used in the threshold integration.

6.14 daind.c

subroutines:
double daind(double *x,double a,double b, double (*fun)(double),double eps,int key,int max).

Main Gaussian integration routine, see [34].

6.15 integrands.c

subroutines:
inline double fcharm_integrand(double x1),
inline double fgluon_integrand(double x1),
inline double fsigma_integrand(double x1),
inline double us_integrand(double x1),
inline double ds_integrand(double x1),
inline double ss_integrand(double x1),
inline double fbottom_integrand(double x1),
inline double light_charm_integrand(double x1).

Functions containing integrands for the threshold integration. They use the density values and the coefficient functions from a-coefs.c to produce the integrands that are then fed into the Gaussian integration program.

6.16 grids.c

subroutines:
int MakeXGrid(void),
int MakeTGrid(void),
int merge(double *a,double *b,int na, int nb,char w),
int check_grid(double *a,int n,char w),
int MakeFortranGrid(int test_mode),
double **allocate_real_matrix(int ur, int uc),
void free_real_matrix(double **m,int ur).

Subroutines for making (and also merging and verifying) the initial grids in x and Q^2 and the final grids for Fortran-code compatible output. The grid merging is used to combine the evenly spaced grid generated automatically from the initial and final values with the premade grid containing several x and Q^2 values for plotting and outputting the data. Two routines are added for deallocating memory.

6.17 weights.c

subroutines:
int readWeights(int nf,int order),
int dumpWeights(int nf,int order).

Routines dealing with loading and saving computed NLO and NNLO weight tables to do a fast calculation on the same grids. LO weights are not saved as it is very fast to compute them every time.

6.18 nnl-a-w.c

subroutines:
int anannlowgts(int nf,int loadWgts),
int computeNNLOwgts(int nf),
double nn_s1ff(double x,double y, int nf),
double nn_s2ff(double x,double y, int nf),
double nn_s1fg(double x,double y, int nf),
double nn_s2fg(double x,double y, int nf),
double nn_s1gf(double x,double y, int nf),

double nn_s2gf(double x,double y, int nf),
double nn_s1gg(double x,double y, int nf),
double nn_s2gg(double x,double y, int nf),
double nn_s1ff_plus(double x, int nf),
double nn_s1gg_plus(double x, int nf),
double nn_s1p(double x,double y, int nf),
double nn_s2p(double x,double y, int nf),
double nn_s1m(double x,double y, int nf),
double nn_s2m(double x,double y, int nf),
double nn_s1p_plus(double x, int nf),
double nn_s1m_plus(double x, int nf),
double nn_s1gf_lim(double sp,double nf),
double nn_s1fg_lim(double sp,double nf),
double nn_s2ff_lim(double sp,double nf),
double nn_s2fg_lim(double sp,double nf),
double nn_s2gf_lim(double sp,double nf),
double nn_s2gg_lim(double sp,double nf),
double nn_s2p_lim(double sp,double nf),
double nn_s2m_lim(double sp,double nf).

Analytically computes or reads from files the approximate NNLO weights for the evolution equations. Here the routines are grouped into three kinds: the nn_s1,2xx routines calculate the regular weights, the nn_s1,2xx_lim routines calculate the regular weights called at 1 and nn_s1,2xx_plus do the weights that contain the plus-distributions.

6.19 wgplg.c

subroutines:
double wgplg(int n,int p,double x).

The routines which calculate polylogarithms using the method from CERNLIB [35]. They are only used for the higher order polylogarithms because the routines for Li2, Li3 and S12 in polylo.c are faster.

7 Results

The code can be used in several modes of operation.

For all of them there is some optimum grid size in x and Q^2. Internally, the grid with the sizes entered in the parameter file is merged with another grid (that is used for plotting the output data at the end), thus increasing the resulting grid size. This internal grid size contains all "popular" values, like $x = 0.1, 0.01, 0.001$ etc., and is 38 in Q^2 and 64 in x. The corresponding values are located in file **main.h** (arrays xpr[] and q2pr[]). This grid is then merged with the automatically generated equidistant grid and the equal values are weeded out. Shown in the table are the resulting grid sizes as shown in the output file. The table uses the calculation for all flavors as opposed to the GRV98-like (only three-flavor) densities. In general, the evolution time grows quadratically in n_x and linearly in n_{Q^2}. The numbers we give below are for an alpha PC with a 21164 processor unit running at 500 MHz, 1 Gbyte of memory and rated at an Specfp = 20.4.

order	n_x	n_{Q^2}	accuracy,digits	time,sec
LO	162	96	5	15
NLO	162	96	3	113
NNLO	162	96	3	385
LO	262	136	6	31
NLO	262	136	5	275
NNLO	262	136	5	1021
LO	362	136	6	44
NLO	362	136	6	529
NNLO	362	136	5	1537

1. Set parameters to the following values to produce LO and NLO GRV98-style fixed three-flavor densities for the whole range of Q^2 (only parameters essential for this calculation are provided, the rest can be set to whatever one wishes since they control the form of the output and similar features, not the physically meaningful ones):

LO	
0.26	Qinitial2
0	Order
1	ThreeFlavorMode
NLO	
0.40	Qinitial2
1	Order
1	AlphaUseExact
1	ThreeFlavorMode

2. To generate regular VFNS densities with all heavy flavors (both charm and bottom) one sets:

LO	
0.26	Qinitial2
0	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
0	NNLOmultiOrderCHARM
1	DoBottomThreshold
NLO	
0.40	Qinitial2
1	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	AlphaUseExact
0	NNLOmultiOrderCHARM
1	DoBottomThreshold

3. To generate VFNS densities involving proper order mixing at heavy thresholds with all heavy flavors (both charm and bottom) but without using NNLO weights (as done in our previous papers [25], [26], [27], [33]) one sets:

LO	
0.26	Qinitial2
0	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	DoBottomThreshold

NLO	
0.40	Qinitial2
1	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	AlphaUseExact
1	DoBottomThreshold

NNLO	
0.40	Qinitial2
2	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	AlphaUseExact
1	NNLOmultiOrderCHARM
1	DoBottomThreshold
1	NLO4NNLO

In this mode it is necessary to generate LO and NLO sets by running the program before running the NNLO set on the same grid! Those will be dumped in special data files

(**agrv99lo.BO.threshold**, **agrv99lo.CH.threshold**,
agrv99nlo.BO.threshold, and **agrv99nlo.CH.threshold**)
that will later be read for the NNLO calculation whenever
NNLOmultiOrderCHARM=1.

4. To generate VFNS densities involving proper order mixing at heavy thresholds with all heavy flavors (both charm and bottom) and using LO, NLO and NNLO (approximate) weights one sets:

LO	
0.26	Qinitial2
0	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	DoBottomThreshold
NLO	
0.40	Qinitial2
1	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	AlphaUseExact
1	DoBottomThreshold
NNLO	
0.40	Qinitial2
2	Order
0	ThreeFlavorMode
1	AlphaDoSeparateThreshold
1	AlphaUseExact
1	NNLOmultiOrderCHARM
1	DoBottomThreshold
0	NLO4NNLO

In this mode it is also necessary to generate LO and NLO sets by running the program before running the NNLO set on the same grid! Those will be dumped in special data files
(**agrv99lo.BO.threshold**, **agrv99lo.CH.threshold**,
agrv99nlo.BO.threshold, and **agrv99nlo.CH.threshold**)
that will later be read for NNLO calculation whenever
NNLOmultiOrderCHARM=1.

Program output is arranged in several forms. First, the default output in normal readable form goes into **resLO.dat, resNLO.dat or resNNLO.dat** or for GRV98-mode into **resLO3.dat, resNLO3.dat or resNNLO3.dat** depending upon the set calculation order. This file contains the input parameters, calculation time and the columns of data versus Q^2 and x for all densities (uv,dv,us,ds,ss,ch and bt, described in previous chapters). Here is the sample:

```
========================= Q2=   1.960 ======================
Alpha(Q2=    1.96 GeV2)=0.318513 for nf=4
------------------------- x=0.000010  ----------------------
SI(x= 0.0000100)=3.4695646e+00  GL(x= 0.0000100)=1.3074834e+01
UV(x= 0.0000100)=6.1120367e-03  DV(x= 0.0000100)=3.7959190e-03
US(x= 0.0000100)=5.9818110e-01  DS(x= 0.0000100)=5.9988948e-01
SS(x= 0.0000100)=5.3175774e-01
CH(x= 0.0000100)=0.0000000e+00  BT(x= 0.0000100)=0.0000000e+00
------------------------- x=0.000020  ----------------------
SI(x= 0.0000200)=3.1153438e+00  GL(x= 0.0000200)=1.1469641e+01
UV(x= 0.0000200)=8.2564168e-03  DV(x= 0.0000200)=5.1210226e-03
US(x= 0.0000200)=5.4149612e-01  DS(x= 0.0000200)=5.4372565e-01
SS(x= 0.0000200)=4.6576141e-01
CH(x= 0.0000200)=0.0000000e+00  BT(x= 0.0000200)=0.0000000e+00
```

The above sample was produced with GraphAll=0 thus printing only values on a small grid with minimum $Q^2 = 1.96$ GeV^2 and not all values from minimum $Q^2 = 0.40$ GeV^2. For convenience, SI denotes singlet, GL gluon, UV and DV are valence densities $u - \bar{u}$, $d - \bar{u}$, US, DS, SS are of the $q + \bar{q} - \Sigma(n_f)/n_f$ kind and CH and BT are $(c + \bar{c})/2$ and $(b + \bar{b})/2$.

For graphing purposes, the output also goes into several datafiles with names formed as **g_densityORDER.dat** where ORDER is LO, NLO or NNLO respectively e.g. **g_glLO.dat** or **g_uvNNLO.dat**. Those contains columns of the particular density with the first column being x or Q^2, depending upon GraphVsX parameter (1-x, 0-Q^2). Then the other parameter is varied across columns. Here is the piece of **g_cpNLO.dat** file. The first column contains the x value, the second is the charm density for $Q^2 = 1.96$ GeV^2 (where it is

zero) and then the charm density for $Q^2 = 2, 3, ..$ GeV^2:

```
0.0000100000 0.0000000000e+00   1.0468420825e-02   1.3022045486e-01
0.0000200000 0.0000000000e+00   8.8559755303e-03   1.0970003578e-01
0.0000300000 0.0000000000e+00   8.0049032415e-03   9.8909559249e-02
0.0000400000 0.0000000000e+00   7.4395488912e-03   9.1758900075e-02
0.0000500000 0.0000000000e+00   7.0219632243e-03   8.6487022224e-02
0.0000600000 0.0000000000e+00   6.6940248888e-03   8.2352079221e-02
0.0000700000 0.0000000000e+00   6.4257535493e-03   7.8973989377e-02
0.0000800000 0.0000000000e+00   6.1998499386e-03   7.6132631291e-02
0.0000900000 0.0000000000e+00   6.0054517691e-03   7.3690103826e-02
```

The above sample was produced with GraphVsX=1 thus printing x, not Q^2 values in the first column. The GraphAll=0 was also set, thus only nice values of x are used (0.00001, 0.00002, 0.00003, etc).

Also, if the necessary option (DoFortran=1) is set the output also goes into the file suitable for reading by a GRV98-like Fortran program that interpolates the data points and makes parton density functions. This program is used in structure function calculations (the code is written in Fortran). The datafile format has eight columns with all densities on the fixed grid (hard-coded into the both evolution code and the interpolation program) in x and Q^2.

The sample follows:

```
Information line: first
+6.112E-03 +3.796E-03 +5.982E-01 +5.999E-01 +5.318E-01 +1.307E+01
+6.128E-03 +3.806E-03 +6.084E-01 +6.101E-01 +5.419E-01 +1.333E+01
+6.303E-03 +3.912E-03 +7.251E-01 +7.268E-01 +6.581E-01 +1.634E+01
+6.440E-03 +3.996E-03 +8.260E-01 +8.278E-01 +7.586E-01 +1.901E+01
+6.553E-03 +4.064E-03 +9.151E-01 +9.169E-01 +8.473E-01 +2.140E+01
+6.649E-03 +4.122E-03 +9.949E-01 +9.967E-01 +9.269E-01 +2.357E+01
+6.731E-03 +4.172E-03 +1.067E+00 +1.069E+00 +9.990E-01 +2.556E+01
+6.804E-03 +4.216E-03 +1.134E+00 +1.135E+00 +1.065E+00 +2.740E+01
+6.869E-03 +4.255E-03 +1.195E+00 +1.197E+00 +1.126E+00 +2.910E+01
+6.927E-03 +4.291E-03 +1.252E+00 +1.253E+00 +1.183E+00 +3.070E+01
```

Sample pictures of bottom densities are provided in [26] and also below in Figs. 1 - 4.

8 Error code descriptions

Program error code description:

message	filename	refer to
Threshold LO datafile is missing	loader.c	NNLO calculation with proper orders requires the datafile from a previous run in LO
Threshold NLO datafile is missing	loader.c	NNLO calculation with proper orders requires the datafile from a previous run in NLO
Wrong Multicharm factor	main.c	NNLOmultiOrderCHARM should be 1 or 0
Wrong order factor	several modules	should be 0,1,2 for LO, NLO, NNLO
File evolution_parameters.input does not exist	main.c	find the file and put into working directory
Wrong INI Q2:increase it!	main.c	order and initial Q^2 are incompatible
Wrong INI Q2:decrease it!	main.c	order and initial Q^2 are incompatible
Evolver: dont know how to proceed	main.c	wrong doBottom factor
Wrong Alpha switch factor!	main.c	check AlphaDoSeparateThreshold value
Wrong order factor while graphing	main.c	check Order to be 0,1,2
Wrong graphing factor	main.c	check GraphAll value to be 0,1
Wrong loadWgts factor	l-a-w.c, nl-a-w.c	check loadWgts to be 0,1

9 Conclusions

We have presented a multifunctional code for the direct x-space method of solving the spin-averaged evolution equations for parton densities. The distinctive features of this code include analytic computation of the LO, NLO and NNLO weights, NNLO heavy flavor threshold matching and NNLO evolution.

The code is very fast and accurate. For example for grid sizes not exceeding 200 in Q^2 and 150 in x the NLO calculation with full weighs computed for three values of n_f and up to five decimal accuracy has a runtime well below 200 seconds. Also it is the only code that does the proper NNLO evolution with NNLO heavy flavor matching conditions.

The program is also easy to use and complete documentation is available. The code is well-tested both on specific test functions (e.g. see [16]) and on actual densities (e.g. see [25]) in all (LO, NLO and NNLO) orders.

10 Acknowledgments

The work was partially supported by the National Science Foundation grant PHY-9722101. We thank Michael Botje for valuable comments on his method of solving the evolution equations and providing us with his evolution code. We also thank Andreas Vogt for help with testing the code in NNLO and for providing comparison data. Thanks are also due to Brian Harris for testing the code and comments on the manuscript.

A Appendix A

Here we give the NNLO parametrizations of the splitting functions from [29]. Note that $L_0 = \ln z$ and $L_1 = \ln(1-z)$.

First the parametrizations for the non-singlet splitting functions $P_{\rm NS}^{(2)\pm}$ are:

$$\begin{aligned}
P_{{\rm NS},A}^{(2)-}(z) =& \, 1185.229\,(1-z)_+^{-1} + 1365.458\,\delta(1-z) - 157.387\,L_1^2 - 2741.42\,z^2 \\
&- 490.43\,(1-z) + 67.00\,L_0^2 + 10.005\,L_0^3 + 1.432\,L_0^4 \\
&+ N_f\,\{-184.765\,(1-z)_+^{-1} - 184.289\,\delta(1-z) + 17.989\,L_1^2 + 355.636\,z^2 \\
&- 73.407\,(1-z)L_1 + 11.491\,L_0^2 + 1.928\,L_0^3\} \;+\; P_{{\rm NS},2}^{(2)}(z), \\
P_{{\rm NS},B}^{(2)-}(z) =& \, 1174.348\,(1-z)_+^{-1} + 1286.799\,\delta(1-z) + 115.099\,L_1^2 + 1581.05\,L_1 \\
&+ 267.33\,(1-z) - 127.65\,L_0^2 - 25.22\,L_0^3 + 1.432\,L_0^4 \\
&+ N_f\,\{-183.718\,(1-z)_+^{-1} - 177.762\,\delta(1-z) + 11.999\,L_1^2 + 397.546\,z^2 \\
&+ 41.949\,(1-z) - 1.477\,L_0^2 - 0.538\,L_0^3\} \;+\; P_{{\rm NS},2}^{(2)}(z), \qquad \text{(A.1)}
\end{aligned}$$

and

$$\begin{aligned}
P_{{\rm NS},A}^{(2)+}(z) =& \, 1183.762\,(1-z)_+^{-1} + 1347.032\,\delta(1-z) + 1047.590\,L_1 - 843.884\,z^2 \\
&- 98.65\,(1-z) - 33.71\,L_0^2 + 1.580\,(L_0^4 + 4L_0^3) \\
&+ N_f\,\{-183.148\,(1-z)_+^{-1} - 174.402\,\delta(1-z) + 9.649\,L_1^2 + 406.171\,z^2 \\
&+ 32.218\,(1-z) + 5.976\,L_0^2 + 1.60\,L_0^3\} \;+\; P_{{\rm NS},2}^{(2)}(z), \\
P_{{\rm NS},B}^{(2)+}(z) =& \, 1182.774\,(1-z)_+^{-1} + 1351.088\,\delta(1-z) - 147.692\,L_1^2 - 2602.738\,z^2 \\
&- 170.11 + 148.47\,L_0 + 1.580\,(L_0^4 - 4\,L_0^3) \\
&+ N_f\,\{-183.931\,(1-z)_+^{-1} - 178.208\,\delta(1-z) - 89.941\,L_1 + 218.482\,z^2 \\
&+ 9.623 + 0.910\,L_0^2 - 1.60\,L_0^3\} \;+\; P_{{\rm NS},2}^{(2)}(z)\;. \qquad \text{(A.2)}
\end{aligned}$$

The parametrizations for $P_{\rm NS}^{(2),S}(z)$ and $P_{\rm PS}^{(2)}(z)$ are

$$\begin{aligned}
P_{{\rm NS},A}^{(2)S}(z) =& \, N_f\,\{(1-z)(-1441.57\,z^2 + 12603.59\,z - 15450.01) + 7876.93\,zL_0^2 \\
&- 4260.29\,L_0 - 229.27\,L_0^2 + 4.4075\,L_0^3\} \\
P_{{\rm NS},B}^{(2)S}(z) =& \, N_f\,\{(1-z)(-704.67\,z^3 + 3310.32\,z^2 + 2144.81\,z - 244.68) \\
&+ 4490.81\,z^2 L_0 + 42.875\,L_0 - 11.0165\,L_0^3\}, \qquad \text{(A.3)}
\end{aligned}$$

and

$$P_{{\rm PS},A}^{(2)}(z) = N_f\,\{(1-z)(-229.497\,L_1 - 722.99\,z^2 + 2678.77 - 560.20\,z^{-1})$$

$$
\begin{aligned}
&+ 2008.61\, L_0 + 998.15\, L_0^2 - 3584/27\, z^{-1} L_0\} \;+\; P_{\rm PS,2}^{(2)}(z), \\
P_{\rm PS,B}^{(2)}(z) = N_f\, \{&(1-z)(73.845\, L_1^2 + 305.988\, L_1 + 2063.19\, z - 387.95\, z^{-1}) \\
&+ 1999.35\, z L_0 - 732.68\, L_0 - 3584/27\, z^{-1} L_0\} \\
&+\; P_{\rm PS,2}^{(2)}(z), \qquad\qquad (A.4)
\end{aligned}
$$

with

$$
\begin{aligned}
P_{\rm PS,2}^{(2)}(z) = N_f^2\, \{&(1-z)(-7.282\, L_1 - 38.779\, z^2 + 32.022\, z - 6.252 + 1.767\, z^{-1}) \\
&+ 7.453\, L_0^2\} \;. \qquad\qquad (A.5)
\end{aligned}
$$

Next we show the parametrizations of the off-diagonal singlet splitting functions:

$$
\begin{aligned}
P_{qg,A}^{(2)}(z) = N_f\, \{&-31.830\, L_1^3 + 1252.267\, L_1 + 1999.89\, z + 1722.47 + 1223.43\, L_0^2 \\
&- 1334.61\, z^{-1} - 896/3\, z^{-1} L_0\} \;+\; P_{qg,2}^{(2)}(z), \\
P_{qg,B}^{(2)}(z) = N_f\, \{&19.428\, L_1^4 + 159.833\, L_1^3 + 309.384\, L_1^2 + 2631.00\, (1-z) \\
&- 67.25\, L_0^2 - 776.793\, z^{-1} - 896/3\, z^{-1} L_0\} \;+\; P_{qg,2}^{(2)}(z), \qquad (A.6)
\end{aligned}
$$

with

$$
\begin{aligned}
P_{qg,2}^{(2)}(z) = N_f^2\, \{&-0.9085\, L_1^2 - 35.803\, L_1 - 128.023 + 200.929\, (1-z) \\
&+ 40.542\, L_0 + 3.284\, z^{-1}\} \;, \qquad\qquad (A.7)
\end{aligned}
$$

and

$$
\begin{aligned}
P_{gq,A}^{(2)}(z) = &\, 13.1212\, L_1^4 + 126.665\, L_1^3 + 308.536\, L_1^2 + 361.21 - 2113.45\, L_0 \\
&- 17.965\, z^{-1} L_0 \;+\; N_f\, \{2.4427\, L_1^4 + 27.763\, L_1^3 + 80.548\, L_1^2 \\
&- 227.135 - 151.04\, L_0^2 + 65.91\, z^{-1} L_0\} \;+\; P_{gq,2}^{(2)}(z), \\
P_{gq,B}^{(2)}(z) = &\, -4.5108\, L_1^4 - 66.618\, L_1^3 - 231.535\, L_1^2 - 1224.22\, (1-z) + 240.08\, L_0^2 \\
&+ 379.60\, z^{-1}(L_0 + 4) \;+\; N_f \{-1.4028\, L_1^4 - 11.638\, L_1^3 + 164.963\, L_1 \\
&- 1066.78\, (1-z) - 182.08\, L_0^2 + 138.54\, z^{-1}(L_0 + 2)\} \\
&+\; P_{gq,2}^{(2)}(z), \qquad\qquad (A.8)
\end{aligned}
$$

with

$$
\begin{aligned}
P_{gq,2}^{(2)}(z) = N_f^2\, \{&1.9361\, L_1^2 + 11.178\, L_1 + 11.632 - 15.145\, (1-z) + 3.354\, L_0 \\
&- 2.133\, z^{-1}\} \;. \qquad\qquad (A.9)
\end{aligned}
$$

Last we show the parametrizations of the diagonal singlet splitting functions

$$
\begin{aligned}
P^{(2)}_{gg,A}(z) =\; & 2626.38\,(1-z)_+^{-1} + 4424.168\,\delta(1-z) - 732.715\,L_1^2 - 20640.069\,z \\
& - 15428.58\,(1-z^2) - 15213.60\,L_0^2 + 16700.88\,z^{-1} + 2675.85\,z^{-1}L_0 \\
& + N_f\,\{-415.71\,(1-z)_+^{-1} - 548.569\,\delta(1-z) - 425.708\,L_1 + 914.548\,z^2 \\
& - 1122.86 - 444.21\,L_0^2 + 376.98\,z^{-1} + 157.18\,z^{-1}L_0\} \\
& + \; P^{(2)}_{gg,2}(z), \\
P^{(2)}_{gg,B}(z) =\; & 2678.22\,(1-z)_+^{-1} + 4590.570\,\delta(1-z) + 3748.934\,L_1 + 60879.62\,z \\
& - 35974.45\,(1+z^2) + 2002.96\,L_0^2 + 9762.09\,z^{-1} + 2675.85\,z^{-1}L_0 \\
& + N_f\,\{-412.00\,(1-z)_+^{-1} - 534.951\,\delta(1-z) + 62.630\,L_1^2 + 801.90 \\
& + 1891.40\,L_0 + 813.78\,L_0^2 + 1.360\,z^{-1} + 157.18\,z^{-1}L_0\} \\
& + \; P^{(2)}_{gg,2}(z), \qquad\qquad \text{(A.10)}
\end{aligned}
$$

with

$$
\begin{aligned}
P^{(2)}_{gg,2}(z) =\; & N_f^2\,\{-16/9\,(1-z)_+^{-1} + 6.4882\,\delta(1-z) + 37.6417\,z^2 - 72.926\,z \\
& + 32.349 - 0.991\,L_0^2 + 2.818\,z^{-1}\}\;. \qquad\qquad \text{(A.11)}
\end{aligned}
$$

B Appendix B

Shown below are the renormalized OME's used for threshold matching calculations in NLO and NNLO (they correspond to the unrenormalized expressions given in Appendix C of [36] and in Appendix A of [30]). All OME'S have been renormalized in the $\overline{\rm MS}$-scheme.

In particular the renormalized coupling α_s is presented in the above scheme for n_f+1 light flavors. Here the heavy quark $H=(c,b)$ is treated on the same footing as the light flavors and it is not decoupled from the running coupling in the VFNS approach. The $(\alpha_s/4\pi)^2$ coefficient in the heavy-quark OME $\tilde{A}_{Hq}^{\rm PS}$ is given by

$$\begin{aligned}
\tilde{A}_{Hq}^{\rm PS,(2)}\left(\frac{m^2}{\mu^2}\right) = C_F T_f \Bigg\{ \Bigg[& -8(1+z)\ln z - \frac{16}{3z} - 4 \\
& + 4z + \frac{16}{3}z^2 \Bigg] \ln^2\frac{m^2}{\mu^2} + \Bigg[8(1+z)\ln^2 z - \left(8 + 40z + \frac{64}{3}z^2\right)\ln z \\
& - \frac{160}{9z} + 16 - 48z + \frac{448}{9}z^2 \Bigg] \ln\frac{m^2}{\mu^2} \\
& + (1+z)\Bigg[32 {\rm S}_{1,2}(1-z) + 16\ln z {\rm Li}_2(1-z) - 16\zeta(2)\ln z \\
& - \frac{4}{3}\ln^3 z \Bigg] + \left(\frac{32}{3z} + 8 - 8z - \frac{32}{3}z^2\right){\rm Li}_2(1-z) \\
& + \left(-\frac{32}{3z} - 8 + 8z + \frac{32}{3}z^2\right)\zeta(2) + \left(2 + 10z + \frac{16}{3}z^2\right)\ln^2 z \\
& - \left(\frac{56}{3} + \frac{88}{3}z + \frac{448}{9}z^2\right)\ln z - \frac{448}{27z} - \frac{4}{3} - \frac{124}{3}z + \frac{1600}{27}z^2 \Bigg\}, \qquad \text{(B.1)}
\end{aligned}$$

The $\alpha_s/4\pi$ and the $(\alpha_s/4\pi)^2$ coefficients of the heavy quark OME's $\tilde{A}_{Hg}^{\rm S}$ are

$$\tilde{A}_{Hg}^{\rm S,(1)}\left(\frac{m^2}{\mu^2}\right) = T_f\left[-4(z^2+(1-z)^2)\ln\frac{m^2}{\mu^2}\right], \qquad \text{(B.2)}$$

and

$$\begin{aligned}
\tilde{A}_{Hg}^{\rm S,(2)}\left(\frac{m^2}{\mu^2}\right) = \Bigg\{ & C_F T_f [(8 - 16z + 16z^2)\ln(1-z) \\
& - (4 - 8z + 16z^2)\ln z - (2 - 8z)]
\end{aligned}$$

$$
\begin{aligned}
&+C_A T_f\Big[-(8-16z+16z^2)\ln(1-z)-(8+32z)\ln z \\
&-\frac{16}{3z}-4-32z+\frac{124}{3}z^2\Big]+T_f^2\Big[-\frac{16}{3}(z^2+(1-z)^2)\Big]\Big\}\ln^2\frac{m^2}{\mu^2} \\
&+\Big\{C_F T_f\Big[(8-16z+16z^2)[2\ln z\ln(1-z)-\ln^2(1-z)+2\zeta(2)] \\
&-(4-8z+16z^2)\ln^2 z-32z(1-z)\ln(1-z) \\
&-(12-16z+32z^2)\ln z-56+116z-80z^2\Big] \\
&+C_A T_f\Big[(16+32z+32z^2)[\mathrm{Li}_2(-z)+\ln z\ln(1+z)] \\
&+(8-16z+16z^2)\ln^2(1-z)+(8+16z)\ln^2 z \\
&+32z\zeta(2)+32z(1-z)\ln(1-z)-\left(8+64z+\frac{352}{3}z^2\right)\ln z \\
&-\frac{160}{9z}+16-200z+\frac{1744}{9}z^2\Big]\Big\}\ln\frac{m^2}{\mu^2} \\
&+C_F T_f\Big\{(1-2z+2z^2)[8\zeta(3)+\frac{4}{3}\ln^3(1-z) \\
&-8\ln(1-z)\mathrm{Li}_2(1-z)+8\zeta(2)\ln z-4\ln z\ln^2(1-z) \\
&+\frac{2}{3}\ln^3 z-8\ln z\mathrm{Li}_2(1-z)+8\mathrm{Li}_3(1-z)-24\mathrm{S}_{1,2}(1-z)] \\
&+z^2\Big[-16\zeta(2)\ln z+\frac{4}{3}\ln^3 z+16\ln z\mathrm{Li}_2(1-z)+32\mathrm{S}_{1,2}(1-z)\Big] \\
&-(4+96z-64z^2)\mathrm{Li}_2(1-z)-(4-48z+40z^2)\zeta(2) \\
&-(8+48z-24z^2)\ln z\ln(1-z)+(4+8z-12z^2)\ln^2(1-z) \\
&-(1+12z-20z^2)\ln^2 z-(52z-48z^2)\ln(1-z) \\
&-(16+18z+48z^2)\ln z+26-82z+80z^2\Big\} \\
&+C_A T_f\Big\{(1-2z+2z^2)[-\frac{4}{3}\ln^3(1-z) \\
&+8\ln(1-z)\mathrm{Li}_2(1-z)-8\mathrm{Li}_3(1-z)]+(1+2z+2z^2) \\
&\times[-8\zeta(2)\ln(1+z)-16\ln(1+z)\mathrm{Li}_2(-z)-8\ln z\ln^2(1+z) \\
&+4\ln^2 z\ln(1+z)+8\ln z\mathrm{Li}_2(-z)-8\mathrm{Li}_3(-z)-16\mathrm{S}_{1,2}(-z)] \\
&+(16+64z)[2\mathrm{S}_{1,2}(1-z)+\ln z\mathrm{Li}_2(1-z)]-\left(\frac{4}{3}+\frac{8}{3}z\right)\ln^3 z \\
&+(8-32z+16z^2)\zeta(3)-(16+64z)\zeta(2)\ln z+(16+16z^2) \\
&\times[\mathrm{Li}_2(-z)+\ln z\ln(1+z)]+\left(\frac{32}{3z}+12+64z-\frac{272}{3}z^2\right)\mathrm{Li}_2(1-z)
\end{aligned}
$$

$$
\begin{aligned}
&-\left(12+48z-\frac{260}{3}z^2+\frac{32}{3z}\right)\zeta(2)-4z^2\ln z\ln(1-z)\\
&-(2+8z-10z^2)\ln^2(1-z)+\left(2+8z+\frac{46}{3}z^2\right)\ln^2 z\\
&+(4+16z-16z^2)\ln(1-z)-\left(\frac{56}{3}+\frac{172}{3}z+\frac{1600}{9}z^2\right)\ln z\\
&-\frac{448}{27z}-\frac{4}{3}-\frac{628}{3}z+\frac{6352}{27}z^2\Bigg\}, \qquad \text{(B.3)}
\end{aligned}
$$

respectively. Now we present the renormalized expressions for the heavy-quark loop contributions to the light-parton OME's denoted by $A_{kl,H}$. The coefficients of the $(\alpha_s/4\pi)^2$ terms in $A_{qq,H}$ and $A_{gq,H}$ are

$$
\begin{aligned}
A_{qq,H}^{\mathrm{NS},(2)}\left(\frac{m^2}{\mu^2}\right) &= C_F T_f\Bigg\{\left[\frac{8}{3}\left(\frac{1}{1-z}\right)_+-\frac{4}{3}-\frac{4}{3}z+2\delta(1-z)\right]\ln^2\frac{m^2}{\mu^2}\\
&+\left[\frac{80}{9}\left(\frac{1}{1-z}\right)_+ +\frac{8}{3}\frac{1+z^2}{1-z}\ln z+\frac{8}{9}-\frac{88}{9}z\right.\\
&\left.+\delta(1-z)\left(\frac{16}{3}\zeta(2)+\frac{2}{3}\right)\right]\ln\frac{m^2}{\mu^2}\\
&+\frac{1+z^2}{1-z}\left(\frac{2}{3}\ln^2 z+\frac{20}{9}\ln z\right)\\
&+\frac{8}{3}(1-z)\ln z+\frac{224}{27}\left(\frac{1}{1-z}\right)_+ +\frac{44}{27}-\frac{268}{27}z\\
&+\delta(1-z)\left(-\frac{8}{3}\zeta(3)+\frac{40}{9}\zeta(2)+\frac{73}{18}\right)\Bigg\}, \qquad \text{(B.4)}
\end{aligned}
$$

and

$$
\begin{aligned}
A_{qq,H}^{\mathrm{S},(2)}\left(\frac{m^2}{\mu^2}\right) &= C_F T_f\Bigg\{\left[\frac{16}{3z}-\frac{16}{3}+\frac{8}{3}z\right]\ln^2\frac{m^2}{\mu^2}\\
&+\left[\frac{160}{9z}-\frac{160}{9}+\frac{128}{9}z+(\frac{32}{3z}-\frac{32}{3}+\frac{16}{3}z)\ln(1-z)\right]\ln\frac{m^2}{\mu^2}\\
&+\frac{4}{3}\left(\frac{2}{z}-2+z\right)\ln^2(1-z)+\frac{8}{9}\left(\frac{10}{z}-10+8z\right)\ln(1-z)\\
&+\frac{1}{27}\left(\frac{448}{z}-448+344z\right)\Bigg\}. \qquad \text{(B.5)}
\end{aligned}
$$

respectively. The coefficients of the $\alpha_s/4\pi$ and $(\alpha_s/4\pi)^2$ terms in $A_{gg,H}$ are

$$A_{gg,H}^{\mathrm{S},(1)}\left(\frac{m^2}{\mu^2}\right) = T_f\left[\frac{4}{3}\delta(1-z)\ln\frac{m^2}{\mu^2}\right], \tag{B.6}$$

and

$$\begin{aligned}
A_{gg,H}^{\mathrm{S},(2)}\left(\frac{m^2}{\mu^2}\right) &= \left\{C_F T_f\left[8(1+z)\ln z + \frac{16}{3z} + 4 - 4z - \frac{16}{3}z^2\right]\right. \\
&+C_A T_f\left[\frac{8}{3}\left(\frac{1}{1-z}\right)_+ + \frac{8}{3z} - \frac{16}{3} + \frac{8}{3}z - \frac{8}{3}z^2\right] \\
&\left.+T_f^2\left[\frac{16}{9}\delta(1-z)\right]\right\}\ln^2\frac{m^2}{\mu^2} \\
&+\left\{C_F T_f\left[8(1+z)\ln^2 z + (24+40z)\ln z - \frac{16}{3z} + 64 - 32z\right.\right. \\
&\left.-\frac{80}{3}z^2 + 4\delta(1-z)\right] + C_A T_f\left[\frac{16}{3}(1+z)\ln z + \frac{80}{9}\left(\frac{1}{1-z}\right)_+\right. \\
&\left.\left.+\frac{184}{9z} - \frac{232}{9} + \frac{152}{9}z - \frac{184}{9}z^2 + \frac{16}{3}\delta(1-z)\right]\right\}\ln\frac{m^2}{\mu^2} \\
&+C_F T_f\left\{\frac{4}{3}(1+z)\ln^3 z + (6+10z)\ln^2 z + (32+48z)\ln z\right. \\
&\left.-\frac{8}{z} + 80 - 48z - 24z^2 - 15\delta(1-z)\right\} \\
&+C_A T_f\left\{\frac{4}{3}(1+z)\ln^2 z + \frac{1}{9}(52+88z)\ln z - \frac{4}{3}z\ln(1-z)\right. \\
&+\frac{1}{27}\left[224\left(\frac{1}{1-z}\right)_+ + \frac{556}{z} - 628 + 548z - 700z^2\right] \\
&\left.+\frac{10}{9}\delta(1-z)\right\},
\end{aligned} \tag{B.7}$$

respectively.

The definitions for the polylogarithms $\mathrm{Li}_n(z)$ and the Nielsen functions $\mathrm{S}_{n,p}(z)$, which appear in the above expressions, can be found in [37].

References

[1] G. Altarelli and G. Parisi, Nucl. Phys. **B126**, 298 (1977); V.N. Gribov and L.N.Lipatov, Sov. J. Nucl. Phys. **15** 438, 675, (1972); Yu. Dokshitser, Sov. Phys. JETP **46**, 641 (1977).

[2] R.G. Roberts, in *The Structure of the Proton,* Cambridge University Press (1993).

[3] R.K. Ellis, W.J. Stirling and B.R. Webber, in *QCD and Collider Physics,* Cambridge University Press (1996), Chapter 4.3.

[4] H. Georgi and H.D. Politzer, Phys. Rev. D**9**, 416 (1974).

[5] D.J. Gross and F. Wilczek, Phys. Rev D**9**, 980 (1974).

[6] E. G. Floratos, D.A. Ross and C.T. Sachrajda, Nucl. Phys. **B129**, 66 (1977) Erratum **B139**, 545 (1978); ibid **B152**, 493 (1979).

[7] A. Gonzales-Arroyo, C. Lopez and F.J. Yndurain, Nucl. Phys. **B153**, 161 (1979); A. Gonzales-Arroyo and C. Lopez, Nucl. Phys. **B166**, 429 (1980).

[8] E.G. Floratos, C. Kounnas and R. Lacaze, Phys. Lett. **B98**, 89, 285, (1981); ibid. Nucl. Phys. **B192**, 417 (1981).

[9] R. Hamberg and W.L. van Neerven, Nucl. Phys. **B359**, 343 (1991).

[10] S.A. Larin, T. van Ritbergen and J.A.M. Vermaseren, Nucl. Phys. **B427**, 41 (1994), [hep-ph/9411260]; S.A. Larin et al., Nucl. Phys. **B492**, 338 (1997), [hep-ph/9605317].

[11] J.F. Bennett and J.A Gracey, Nucl. Phys. **B417**, 241 (1998); J.A. Gracey, Phys. Lett. **B322**, 141 (1994), [hep-ph/9401214].

[12] W.L. van Neerven and A. Vogt, [hep-ph/9907472].

[13] M. Miyama and S. Kumano, Comput. Phys. Commun. **94**, 185 (1996), [hep-ph/9508246]; M. Hirai, S. Kumano and M. Miyama, Comput. Phys. Commun. **108**, 38 (1998), [hep-ph/9707220].

[14] M. Botje, QCDNUM16: A fast QCD evolution program, ZEUS N5te 97-066.

[15] C. Pascaud and F. Zomer, H1 Note H1-11/94-404; V. Barone, C. Pascaud and F. Zomer, [hep-ph/9907512].

[16] J. Blümlein, S. Riemersma, W.L. van Neerven and A. Vogt, Nucl. Phys. **B**(Proc. Suppl.) **51C**, 96 (1996), [hep-ph/9609217];
J. Blümlein, M. Botje, C. Pascaud, S. Riemersma, W.L. van Neerven, A. Vogt and F. Zomer, in *Proceedings of the Workshop on Future Physics at HERA* edited by G. Ingelman, A. De Roeck and R. Klanner, Hamburg, Germany, 25-26 Sep. 1995, p. 23, DESY 96-199, [hep-ph/9609400].

[17] A.D. Martin, R.G. Roberts, W.J. Stirling and R. Thorne, Eur. Phys. J. **C4**, 463 (1998), [hep-ph/9803445].

[18] H.L. Lai, J. Huston, S. Kuhlmann, J. Morfín, F. Olness, J. Owens, J. Pumplin and W.K. Tung, [hep-ph/9903282].

[19] G. Curci, W. Furmanski and R. Petronzio, Nucl. Phys. **B175**, 27 (1980); W. Furmanski and R. Petronzio, Phys. Lett. **B97** 437, (1980); ibid. Z. Phys. **C11**, 293 (1982); the relevant NLO formulae are presented in a convenient form in ref. 2.

[20] C. Coriano and S. Savkli, Comput. Phys. Commun. **118**, 236 (1999), [hep-ph/9803336].

[21] S. Riemersma, unpublished.

[22] M. Glück, E. Reya and A. Vogt, Eur. Phys. J. **C5**, 461 (1998), [hep-ph/9806404].

[23] H. Plothow-Besch, PDFLIB version 8.04, available from CERNLIB at http://wwwinfo.cern.ch/asdoc.

[24] M.A.G. Aivazis, J.C. Collins, F.I. Olness and W.-K. Tung, Phys. Rev. D**50**, 3102 (1994), [hep-ph/9312319].

[25] A. Chuvakin, J. Smith and W. van Neerven, Phys. Rev. D**61**, 096004 (2000), [hep-ph/9910250].

[26] A. Chuvakin, J. Smith and W. van Neerven, Phys.Rev. **D62**, 036004 (2000), [hep-ph/0002011].

[27] A. Chuvakin, J. Smith and B.W. Harris, Eur.Phys.J. C**18**,547 (2001), [hep-ph/0010350].

[28] W.L. van Neerven; A. Vogt, [hep-ph/9907472].

[29] W.L. van Neerven; A. Vogt, [hep-ph/0007362].

[30] M. Buza, Y. Matiounine, J. Smith and W.L. van Neerven, Eur. Phys. J. **C1**, 301 (1998); Phys. Lett. B**411**,211 (1997), [hep-ph/9612398].

[31] W. Bernreuther and W. Wetzel, Nucl. Phys. **B197**, 228 (1982); Erratum-ibid **513**, 758 (1998); W. Bernreuther, Annals of Physics, **151**, 127 (1983).

[32] S.A. Larin, T. van Ritbergen and J.A.M. Vermaseren, Nucl. Phys. **B438**, 278 (1995), [hep-ph/9411260]; see also K.G. Chetyrkin, B.A. Kniehl and M. Steinhauser, Phys. Rev. Lett. **79**, 2184 (1997), [hep-ph/9706430].

[33] A. Chuvakin, J. Smith, Phys. Rev. D**61**, 114018 (2000); [hep-ph/9911504].

[34] R. Piessens, Angew. Informatik **9**, 399 (1973).

[35] K.S. Kölbig, J.A. Mignaco and E. Remiddi, BIT **10**, 38 (1971);
K.S. Kölbig, SIAM J. Math. Anal. **17** 1232 (1986);
see http://wwwinfo.cern.ch/asdoc/shortwrupsdir/c31/top.html.

[36] M. Buza, Y. Matiounine, J. Smith, R. Migneron, and W.L. van Neerven, Nicl. Phys. **B472**, 611 (1996), [hep-ph/9601302].

[37] L. Lewin, "Polylogarithms and Associated Functions", North Holland, Amsterdam, 1983;
R. Barbieri, J.A. Mignaco and E. Remiddi, Nuovo Cimento **11A** (1972) 824;
A. Devoto and D.W. Duke, Riv. Nuovo. Cimento Vol. 7,N. 6 (1984) 1.

Figure Captions

Fig. 1. The gluon density $xg_{\rm NNLO}(4,x,\mu^2)$ in the range $10^{-5} < x < 1$ for $\mu^2 = 2$, 3, 4, 5, 10 and 20 in units of $({\rm GeV}^2)^2$,

Fig. 2. The singlet density $x\Sigma_{\rm NNLO}(4,x,\mu^2)$ in the range $10^{-5} < x < 1$ for $\mu^2 = 2$, 3, 4, 5, 10 and 20 in units of $({\rm GeV}^2)^2$,

Fig. 3. The nonsinglet quark density $x\sigma_{\rm NNLO}(4,x,\mu^2)a$, where $\sigma = (u+\bar{u})/2$, in the range $10^{-5} < x < 1$ for $\mu^2 = 2$, 3, 4, 5, 10 and 20 in units of $({\rm GeV}^2)^2$,

Fig. 4. The charm quark density $xc_{\rm NNLO}(4,x,\mu^2)$ the range $10^{-5} < x < 1$ for $\mu^2 = 1.96$, 2, 3, 4, 5, 10 and 20 in units of $({\rm GeV}^2)^2$,